WL-TM-96-1134

ACHIEVING NEAR-OPTIMAL SENSOR
ALLOCATION POLICIES THROUGH REINFORCEMENT
LEARNING

RAJ P. MALHOTRA

OCTOBER 1996

FINAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

19961125 144

AVIONICS DIRECTORATE
WRIGHT LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT PATTERSON AFB OH 45433-7623

REPORT DOCUMENTATION PAGE

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE OCT 1996	3. REPORT TYPE AND DATES COVERED FINAL

4. TITLE AND SUBTITLE ACHIEVING NEAR-OPTIMAL SENSOR ALLOCATION POLICIES THROUGH REINFORCEMENT LEARNING

6. AUTHOR(S) RAJ P. MALHOTRA

5. FUNDING NUMBERS
C
PE   61102
PR   2304
TA   ES
WU   01

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

AVIONICS DIRECTORATE
WRIGHT LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT PATTERSON AFB OH 45433-7623

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AVIONICS DIRECTORATE
WRIGHT LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT PATTERSON AFB OH 45433-7623

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

WL-TM-96-1134

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

TACTICAL AIRCRAFT MUST FREQUENTLY PERFORM COMPLEX SEQUENTIAL TASKS IN WHICH THEY RELY HEAVILY ON THE INTEGRATION OF SENSORY DATA TO ASSESS STATE AND MAINTAIN SITUATIONAL AWARENESS. IN MODERN SYSTEMS, THE CONTROL OF THE SENSORS' INFORMATION-GATHERING ACTIVITIES IS CRITICAL-OPTIMAL PERFORMANCE IS DESIRED. BUT THIS IS MADE DIFFICULT BY THE REQUIREMENTS TO CONTEND WITH SOPHISTICATED FLEXIBLE SENSORY ASSETS, AND VOLATILE, UNCERTAIN ENVIRONMENTS. THIS PAPER INTRODUCES THE SENSOR MANAGEMENT PROBLEM AND THE PLAUSIBILITY OF LEVERAGING A MACHINE LEARNING ALGORITHM TOWARD THIS DIFFICULT CHALLENGE.

14. SUBJECT TERMS			15. NUMBER OF PAGES 8
			16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT SAR

DTIC QUALITY INSPECTED 3

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to *stay within the lines* to meet *optical scanning requirements*.

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	-	Contract	**PR** - Project	
G	-	Grant	**TA** - Task	
PE	-	Program Element	**WU** -	Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

> DOD - See DoDD 5230.24, "Distribution Statements on Technical Documents."
> DOE - See authorities.
> NASA - See Handbook NHB 2200.2.
> NTIS - Leave blank.

Block 12b. Distribution Code.

> DOD - Leave blank.
> DOE - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
> NASA - Leave blank.
> NTIS - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

ACHIEVING NEAR-OPTIMAL SENSOR ALLOCATION POLICIES THROUGH REINFORCEMENT LEARNING

Raj Malhotra

Wright Laboratory, Avionics Directorate
2144 Avionics Circle
WPAFB,OH 45433

Abstract--Tactical aircraft must frequently perform complex sequential tasks in which they rely heavily on the integration of sensory data to assess state and maintain situational awareness. In modern systems, the control of the sensors' information-gathering activities is critical--optimal performance is desired. But this is made difficult by the requirements to contend with sophisticated/flexible sensory assets, and volatile, uncertain environments. This paper introduces the sensor management problem and the plausibility of leveraging a machine learning algorithm toward this difficult challenge.

INTRODUCTION

In the past, many of the most critical parts of the sensor integration task onboard military aircraft were performed by the pilot. This included the moding and pointing of sensors as well as the interpretation of the resulting data to provide a basis for future resource allocation decisions. But factors such as the increased sophistication of avionics assets and the intricate nature of the sensory environment have pushed this task beyond the capability of human operators and motivated a relatively new science, Sensor Fusion[1]. The process of Sensor Fusion involves gathering sensory data, refining and interpreting it, and making new sensor allocation decisions. The sensor allocation process, sensor management, is believed to possess many of the hardest attributes of any scheduling or control problem known in the open literature and is our focus here. We shall motivate the problem with two examples.

Example 1 - Search vs. Track vs. ID

Imagine that we are piloting a tactical fighter, perhaps in a counter-air mission in which it is crucial that we maintain awareness of all aircraft in a vicinity extending beyond visual range (see figure 1). Our fighter has onboard an ideal

---

1. This new science may also be called Data Fusion or Information Fusion--the terminology for this area has not been standardized.

data fusion system; given a particular sequence of measurements, the error in its state assessments are due entirely to the insufficiencies of the input measurements, themselves. Our challenge is to specify a sequence of measurements to
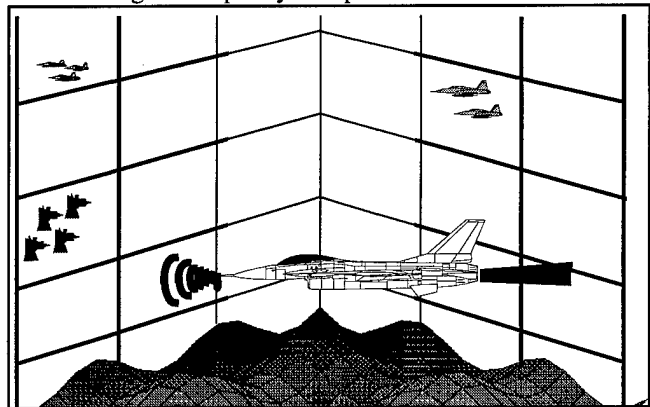

Figure 1 - Search vs. Track vs. Identification

drive this data fusion system. At the current time, we are aware of some objects in the distance, but have not been able to clearly establish their number, position, or identity (ID). We are further aware of two enemy aircraft and their positions--since these aircraft are known hostiles we shall want to keep accurate estimates of their kinematics so that we (or others) may engage them. Lastly, we have recently received intelligence reports that a group of hostile aircraft may be approaching from beyond sensory range--we are uncertain about where they may appear. Thus we have three requirements;

- to search the sectors from which new threats may appear;
- to track known hostile aircraft;
- to resolve the number and identity of some suspected objects in the far distance.

To achieve our mission objectives, we must balance these three jobs (search, track, and ID) amidst the complexities/limitations of our sensor suite, our avionics, and our operating environment. For this example we shall assume we have only a single sensor, an electronically scanned aperture (ESA) radar which can be moded and pointed to resolve

uncertainties in the presence, identity, and the position of objects. We shall need to consider that there is a tradeoff between the time and energy we spend on an object and the degree to which we can resolve its kinematics or identity.

Suppose that we choose to prioritize the candidate measurements according to some well-defined, pre-computed metric. At a particular point in time, our calculations indicate that the most profitable sequence of measurements would be first to resolve the identity of the group of objects in the distance, followed by spending equal amounts of time/energy alternating between tracking the two hostile aircraft and searching for the group of hostiles believed to approaching. During the execution of the first task, the two hostile aircraft begin evasive maneuvers in preparation for an attack. Upon establishing the identity and number of objects in the distance we attempt to update our tracks on the two hostiles only to discover that their position (and kinematics) have changed significantly since our last observation. This degrades the quality of our current measurement and consequently our ability to engage these two targets. Even if we are able to adapt by re-prioritizing activities to refine tracks first, we have lost some degree of effectiveness and depending on future events (arrival of new hostiles, actions of attacking aircraft, etc...) this may change the outcome of our entire mission.

Example 2 - Threat vs. Opportunity

In the second example (illustrated in figure 2) our mission is to find and destroy important, mobile enemy ground assets such as disguised rocket-launchers or tanks. We are equipped the following sensors (and constraints):

- an ESA, provides high-quality kinematic and medium-quality ID information, cannot operate simultaneously with ESM, for air-to-air (ATA) applications
- an Electronic Surveillance Measure (ESM), provides high-quality ID but low-quality kinematic information, cannot operate simultaneously with ESA or SAR, ATA application
- Synthetic Aperture Radar (SAR), provides medium-quality ID and kinematic information, interferes with ESM, air-to-ground (ATG) application
- Forward-Looking Infrared (FLIR) sensor, provides high-quality ID, azimuth, and elevation information, ATG application

Upon identifying targets with our SAR in a low-resolution (wide field-of-view) mode, we wish to obtain high-resolution images near the targets by using our SAR in a high-reso-

lution (spotlight) mode. This data will then be used to cue our FLIR to obtain improved ID and to support targeting. Since the targets are disguised and mobile, we shall need to perform the SAR-FLIR-targeting sequence contiguously and in a short amount of time. Suppose that during the first part of this sequence we become aware of an incoming airborne
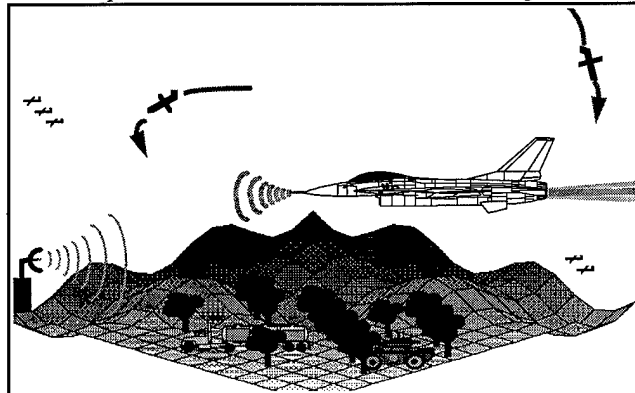

Figure 2 - High-Stress Scenario

object which may be a threat. Depending on the identity, position, and intent of this object, it may be critical to ID it or resolve kinematics. The ESA produces superior spatial fixes, while the ESM produces superior ID. Which should we use first? It is a question concerning both the relative quality and value of information involved. But we have still another, perhaps more critical, decision. The degree to which we engage the airborne object will have an impact on our ability to target and destroy important ground-based threats. Should we abandon the targeting sequence against the ground threats? Ultimately, this will depend upon our preferences for taking risks to achieve mission objectives. We shall need to perform a complex tradeoff between the estimated opportunity of the ATG targeting sequence and the estimated criticality of the inbound target. This requires reasoning over uncertain quantities such as estimated time of completion of the ATG sequence, the IDs and kinematics of airborne and ground-based objects, and the overall tactical situation.

The aforementioned examples illustrate certain fundamental facts about sensor management:

- decisions are complex; they depend upon numerous, uncertain and interrelated quantities
- decisions must be made in real-time; in high-stress scenarios there is little time to compute solutions potentially pushing SMgmt beyond the capability of human operators
- measurements are noisy and occur as point estimates in a larger picture, whose context can only be discerned over extended time-horizons

These facts have motivated the study of automated sensor managers which can help the pilot deal with high- stress scenarios involving heavy enemy defenses, many cooperating

platforms, low observable targets, and high clutter environments [10].

MARKOV DECISION PROCESSES

Sensor management can be viewed as a sequential decision problem in which the penalties and rewards for actions are only revealed over time. In this section we briefly discuss the primary model for such problems, a Markov Decision Process (MDP).

A dynamic process, described by a finite number of states, is said to be Markovian when the state of the process at any given time is solely a function of the state and inputs at the previous time.[1] This may be expressed as

$$x(t_{i+1}) = f(x(t_i), u(t_i), w(t_i), t_i)$$

where x(t) denotes process state, u(t) denotes controllable inputs, and w(t) denotes a disturbance. Note also that the dynamic propagation function, f(-), may, in general, be time-variant. We further stipulate that there is a cost function, g(x(t),u(t),t) associated with states and actions. The cost functions will be derived from some measures of performance (as discussed above).

For SMgmt, the MDP is used to represent the evolution of a model through time. At each step, we are in a particular state, x(t_i), and faced with a set of control alternatives, {u(t_i)}. In figure 7 below, there are costs associated with each state (circle) and each alternative action (arc) as indicated. Our goal is to find a trajectory to minimize the cost incurred as we traverse the MDP from the initial state to the terminal state. The numbers in boxes above the states represent minimum cost-to-go achievable. We may contrast the performance achieved when a predictive approach is used (thickest arcs) with that obtained for a greedy approach (gray arcs). For the greedy approach we pick the alternative with the least cost over one time step (arc cost plus next state cost). The predictive approach is able to derive the minimum cost-to-gos for the states of the next step and achieve optimal performance (28) by always traveling to the state with the least cost-to-go[2]. In contrast, the greedy approach achieves a suboptimal performance (32).
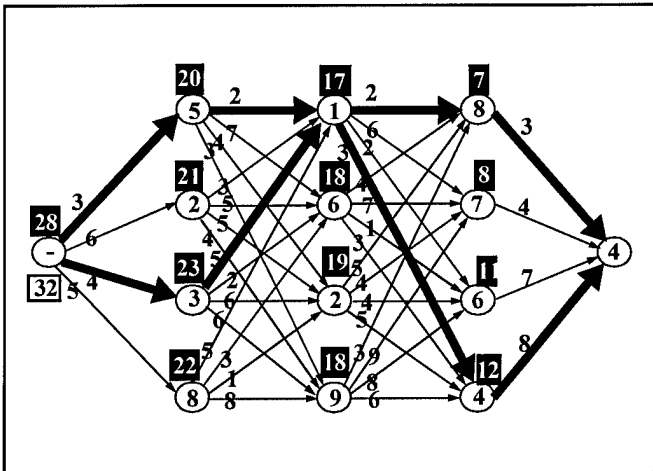

Figure 7 - A Deterministic MDP

To model SMgmt we shall require significantly more complex MDPs than as shown in figure 7. First, SMgmt is stochastic, results of actions and state costs are uncertain. Second, there will be minor variations in small costs (perhaps due to variations in entropy) until some relatively significant event occurs (loss of track, failure to detect hostile in time, etc.). Third, there is no terminal or goal state in SMgmt: we continue sensing over a virtually infinite horizon. Finally, there will be many more states at each time step. Figure 8 incorporates these factors to illustrate the form of a more realistic MDP The variations in the darkness of the arcs indicate their relative likelihood of occurrence. Also, all costs are mean values of cost probabilities; there will also be higher order moments (variances) associated with these costs. We may observe that the optimal cost-to-go from state
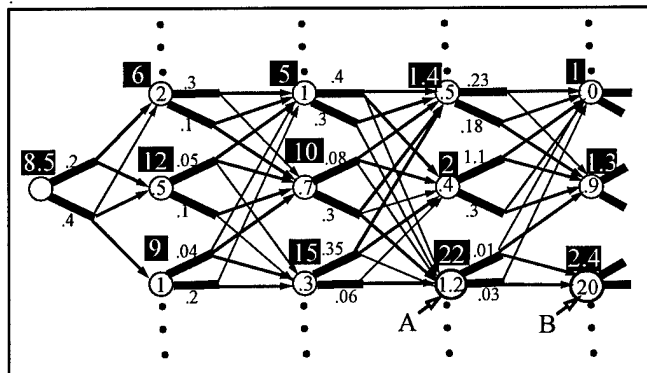

Figure 8 - A Stochastic, Infinite-Horizon MDP

A is quite high due to a catastrophic event in state B. But the stochastic nature of the MDP makes it much more difficult to plan how to avoid these statse.

---

1. We assume a discrete-time process here although continuous-time processes may also be Markovian.
2. This is the *Principle of Optimality*, the basis of the Dynamic Programming approach.

TEMPORAL DIFFERENCE LEARNING

Fortunately, problems quite similar to sensor management are encountered in a wide variety of interesting engineering situations. In heuristic search applications one attempts to learn an evaluation function which predicts the utility of searching in a certain region of the search space. Game-like decision processes involve learning to predict outcomes based upon current state. Difficult modeling tasks and pattern recognition problems often involve learning process complexities through observing sequential behavior. The common thread in these involves *learning to predict* process outcomes and sequential behavior. Often, the goal of prediction is not an end in itself, but rather a prerequisite for effective control of some complex, sequential system.

For problems in which apriori knowledge is limited, unreliable, or unavailable, a machine-learning methodology has developed. *Reinforcement learning* methods ([17],[12],[13]) have received increasing attention as viable control mechanisms which can achieve near-optimal performance. Mathematically, these algorithmic methods may be viewed as a form of Iterative, Stochastic Dynamic Programming in which one attempts to approximate value iteration (or policy iteration) as process characteristics are learned through experience. The hallmark of reinforcement learning methods is their treatment of actual, experienced state transitions and rewards as unbiased estimators of the statistically true values. This provides for incremental calculations which have been shown to theoretically converge in the limit to optimal predictions.

Temporal Difference Learning [12] has received increased attention in recent years as a type of reinforcement learning which learns to predict outcomes of markov processes with delayed and accrued rewards. The significance of this method is that it can learn to address the difficult temporal credit assignment problem in which one must assign credit (or blame) to decisions when the feedback is noisy and delayed. This is believed to describe many real-world situations in which more conventional, supervised learning methods are less efficient (or not viable) because of the reliance on the availability of accurate feedback at each decision point. In the absence of such feedback, the task of calculating optimal actions at each decision point becomes formidable. Whereas conventional prediction-learning methods assign credit (value of decisions) by means of the difference between predicted and actual outcomes, Temporal Difference Learning assigns credit by means of the difference between temporally successive predictions. This allows for more efficient learning.

The original motivation for the development of temporal difference learning methods involved the *temporal credit assignment* problem. There, one experiences a series of state transitions which terminate when an *absorbing state* is reached. There is a cost, z, associated with the terminal state; all other states (and transitions) are nominally assumed to have zero cost. The goal of the Temporal Difference Learner is to derive a sequence of predictions $\{P_1,P_2,...,P_n\}$ of the final outcome based upon a sequence of experienced states, $\{x_1,x_2,...\}$ (as shown in figure 9).
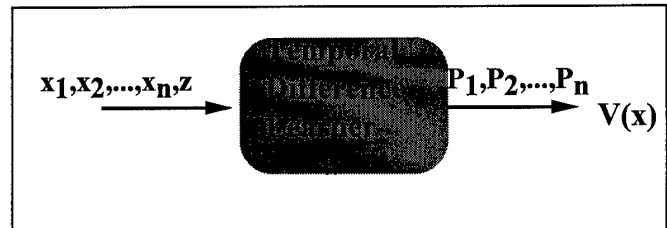

Figure 9 - Conceptual depiction of TDL

The sequence of states is assumed to be a markov process in which state transition probabilities are dependent only upon the most recent state: $p(x_{t+1}|x_t,x_{t-1},x_{t-2},...,x_1) = p(x_{t+1}|x_t)$. The learning agent is parameterized by a vector of weights, w, which control the sequence of outcome predictions based upon state trajectories. The goal is to find the appropriate weight settings so that the prediction values are optimized. The optimal prediction values are known to satisfy the Bellman Equation:

$$P_w^*(x) = \min/\max[R(x) + \gamma \sum_{x'} p(x'|x)P_w^*(x) \quad \textbf{(1)}$$

where R(x) is the incremental transition cost to leave state x, x' is the state transitioned to, P_w^* is the weight-parameterized optimal prediction value, and γ is a discount factor in (0,1). The learning procedure is expressed as a rule for updating w:

$$w(n+1) = w(n) + \sum_{t=1}^{m} \Delta w(t) \quad \textbf{(2)}$$

In practice the change in weights,Δw, may be accrued over a complete sequence, until a termination is reached. A typical supervised learning method would associate each state with the final outcome and perform gradient-descent based upon each pair separately:

$$\Delta w(t) = \alpha(z - P_t) \{\nabla_w P_t\} \quad \textbf{(3)}$$

where α is an update rate effecting the rate of convergence and the gradient, $\nabla_w P_t$, is a vector of partial derivatives of P_t with respect to w. The difficulty with this approach is that the error terms, $(z-P_t)$, can only be computed at the end of each sequence, when an outcome is achieved. This is

computationally expensive. The temporal difference method is based upon equating the final prediction errors to a string of intermediate prediction errors which can be computed incrementally:

$$z - P_t = \sum_{k=t}^{m} P_{k+1} - P_k \;\leftrightarrow\; E(z) \approx P_{k+1} \quad \textbf{(4)}$$

A key point is that the individual predictions serve as unbiased estimates of the final outcome, z. By substituting equation (4) into equations (3) and (2) we derive a new weight update equation:

$$w \leftarrow w + \sum_{t=1}^{m} \alpha\big(z - P_t\big)\nabla_w P_t$$

$$= w + \sum_{t=1}^{m} \alpha\left(\sum_{k=t}^{m}\big(P_{k+1} - P_k\big)\right)\nabla_w P_t \quad (5)$$

$$= w + \sum_{t=1}^{m} \alpha(P_{t+1} - P_t)\sum_{k=1}^{t}\nabla_w P_k$$

where m represents the number of state transitions experienced in a particular *markov chain*--this will in general be a random variable. Note the last summation term in (5); this is an *eligibility* term which assigns credit over a string of past states. Finally, note that (5) can be computed incrementally, after each state transition. Weight updates may either be performed after the markov chain has completed (*batch* mode) or alternatively, after each individual state transition (*on-line* learning).

The TD(λ) family of learning procedures modifies equation (5) by the inclusion of an exponential weighting factor on the eligibility, λ ε [0,1]. This results in a new weight update procedure in which the incremental weight changes are given by

$$\Delta w(t) = \alpha\big(P_{t+1} - P_t\big)\sum_{k=1}^{t} \lambda^{t-k}\nabla_w P_t \quad (6)$$

Equation (6) may be thought of as an error term for temporal credit assignment coupled with a gradient-descent mechanism for structural credit assignment. The last summation term can be thought of as a state eligibility factor, e_t, given by

$$e_{t+1} = \sum_{k=1}^{t+1} \lambda^{t+1-k}\nabla_w P_k = \nabla_w P_{t+1} + \sum_{k=1}^{t} \lambda^{t+1-k}\nabla_w P_k$$

$$= \nabla_w P_{t+1} + \lambda e_t \quad (7)$$

Note that this eligibility factor may also be computed recursively.

Using the equations given above, a synopsis of the TD(λ) learning algorithm is given below:
  1. Initialize parameters; x, w, λ, α
  2. Observe a state transition
  3. Compute new prediction, $P_t = f(x_t, w_t)$
  4. Compute the new eligibility, e_{t+1}
  5. Compute incremental weight update via (6)
  6. If x_t is not an absorbing state, Go To step 2

The incremental weight updates computed in step 5 may either be immediately applied to the weights (in the case of on-line learning) or they may be summed and applied to the weights after the completion of a markov chain (in the case of batch mode learning).

Using the temporal difference learning algorithm we may attack the sensor management problem posed as a markov decision process. The basic approach is to use TDL to predict the outcomes for various alternative actions and to choose the action with the most favorable predicted outcomes. Several simulations which show the preliminary promise of TDL for sensor management are given below.

SIMULATION RESULTS

In order to explore the plausibility of using TDL for sensor management, we performed several simulations at a somewhat abstract level. The results presented herein suggest that machine learning may be useful for this problem but clearly, more work must be done to take the application from the abstract to the specific, detailed scenarios--only then can we accurately evaluate the applicability of this technology.

We shall document two experiments in which the sensor management challenge was modeled as a markov decision process. This was accomplished by establishing a two-dimensional 7x7 grid in which the *ownship* occupies the center position and remains stationary while (possibly hostile) objects move about the grid. The objects are always introduced from the edges of the grid and are of one of several varieties:
- Friendly - These objects do not carry/fire weapons
- Foe 1 - moderately effective weapons, moderate speed
- Foe 2 - highly effective weapons; high speed aircraft

Similar to the objects above, the ownship carries a weapon which may be used to destroy objects in the grid. But unlike

the other objects, the ownship does not have a perfect view of the world and can only see through its single sensor which takes noisy measurements of the grid. A conceptual picture of the grid and the confusion matrix, giving probabilities of detection and false alarm for a signal-to-noise ratio of 1 are given in figure 10. The input into the temporal difference learning algorithm is a state vector which includes hypotheses for the different objects in each grid cell (see figure 10). These hypotheses are adjusted with each new measurement (assuming exclusivity between cells) by using Bayes Rule:
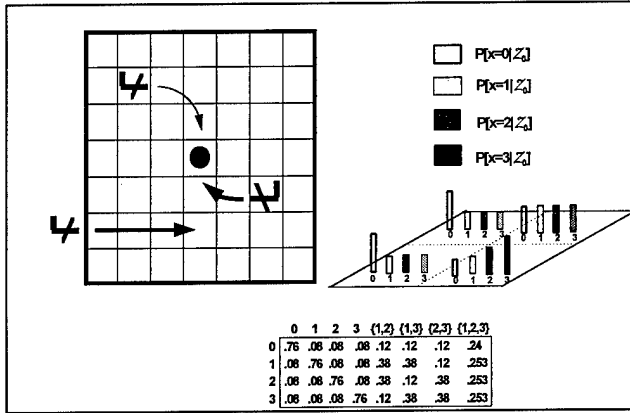

Figure 10 - simulation model

$$P_{k+1}(x = 1|Z_k) = \frac{P(z_k|x = 1)P_k(x = 1)}{\sum_{x \in X} P(z|x)P(x)} \quad (8)$$

where k indexes iterations, x_k is the state vector, Z_k is the measurement history vector, and z_k is a recent individual measurement. The state vector also consists of a particular action to be taken out of the current state. The TDL takes a history of the past 4 state vectors and produces a prediction of the expected outcome (in our simulation this also corresponds to the probability of survival). This is shown conceptually in figure 11 below.
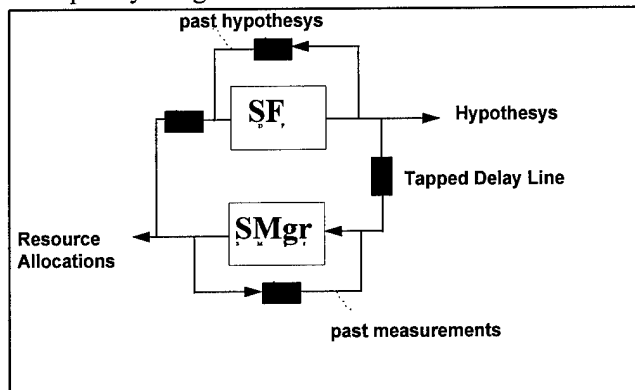
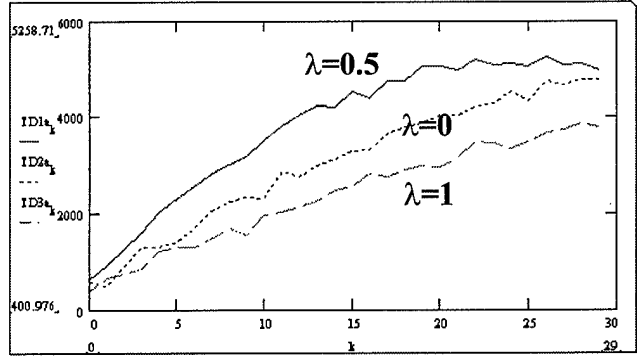

Figure 11 - simulation model

Figure 12 - Performance of TDL

In the first sets of simulations, TDL was used with a standard Backpropagation neural network (for function approximation). Thirty epochs (sets of trials) consisting of 50 trials each were generated and learned over. The results were averaged over 10 training sessions and are presented in figure 12 below. This figure shows the time of survival (in simulation time steps) versus several different settings for the "lookahead" parameter, λ. Thess simulations were performed with an "influx probability" of 0.1 (this is the probability of a new arrival on each time step. The purpose of this simulation was to first verify that TDL could learn to perform better over repeated trials and secondly, to examine the tradeoff in and sensitivity to λ. The results show that after a lengthy amount of training TDL does improve its performance and that its performance is quite sensitive to λ This indicates that the parameters for the learning algorithm must in general be chosen with some care.

Upon establishing that TDL can learn to improve its performance over time, the next question becomes "*How well does TDL perform relative to optimal performance?*" This question motivated a second set of simulations in which the scenario was controlled to be one in which an optimal policy was offered by the theory. In particular a linear-quadratic model was used in which the state followed the propagation equation

$$x_{k+1} = Ax_x + Bu_k + w_k \quad (9)$$

and the objective is to minimize the quadratic cost functional

$$\underset{\substack{w_k \\ k=0,\dots,N-1}}{E} \left\{ x_N Q_N x_N + \sum_{k=0}^{N-1} \left(x_k' Q_k x_k + u_k' R_k u_k \right) \right\} \quad (10)$$

with the matrices A,B,Q, and R all required to be positive semi-definite [8]. Under these conditions, it is a well-known

result of stochastic dynamic programming that a closed form optimal solution exists and is given by

$$\mu_k^*(x_k) = L_k x_k$$

where

$$L_k = -(B_k'K_{k+1}B_k + R_k)^{-1}B_k'K_{k+1}A_k \qquad (11)$$

K_i are given by

$$K_N = Q_N$$

$$K_k = A_k'\left[K_{k+1} - K_{k+1}(B_k'K_{k+1}B_k + R_k)^{-1}B_k'K_{k+1}\right]A_k + Q_k$$

The linear-quadratic (LQ) model is similar to a regulation problem in which one would like to keep the state vector close to zero (or some nominal value or trajectory) and receives relatively large penalty for deviating significantly from that goal while receiving smaller costs for small deviations. This is a reasonable represenatation for many situations (although for sensor management the linear state propagation function is questionable).

The significance of the LQ model is that it offers us a "yardstick" with which we can measure the performance of our learning algorithm's performance. The closed-form solution given in equation 11 is provably optimal. In figure 13 below we compare the performance of the optimal solution (marked "DP" for Dynamic Programming) against that of two differing versions of the learning algorithm. The curve marked "TDL" is standard temporal difference
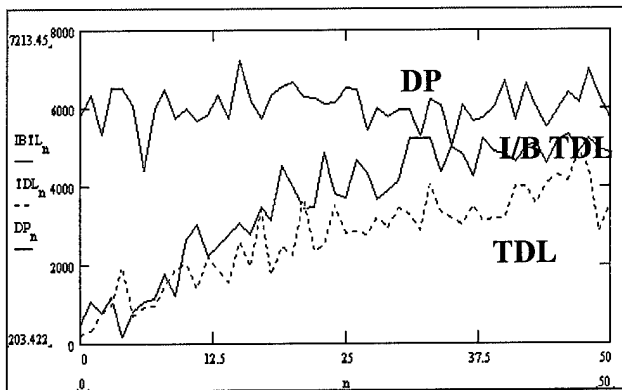

Figure 13 - Comparative results for LQ case

learning with backpropagation for state estimation. The state vector presented to the learning agent is simply the grid of hypothesys (with tapped delay lines giving the last 4 values) as described above. In the curve marked "I/B TDL" we present the learner with an immanence-based state vector which represents the risk to the learning agent by only encoding the values of the proximity of the objects and the risk that they pose (risk factor = Σ (object risk * object belief). This encoding of the state vector produces a state

space of smaller cardinality and allows one to employ Linear Temporal Difference Learning in which no function approximation scheme, such as backpropagation, is necessary--each state value corresponds to a unique weight in the learning agent (thus there is no interference between weights when learning). We may observe from figure 13 that this formulation performs better (learns faster) than standard TDL with backpropagation. This result suggests that the performance of TDL (or any learning scheme) will be somewhat dependent upon the internal state representation presented to the learning agent. The investigation of this phenomenon is important since it will help to identify how TDL can most effectively scale up to handle the more complex, large state spaces of the full sensor management problem.

CONCLUSION

We have briefly discussed and demonstrated the ability of Temporal Difference Learning to address the sensor management challenge. Although the results are for a simple simulation, we believe machine learning algorithms do show promise for this problem. In particular, they allow one to leverage off-line processing (in the form of training the learner) toward a difficult time-constrained problem in which one hopes to find near-optimal sensor allocations. The ability of these learning algorithms to scale is a chief concern for the future work in this area.

REFERENCES

[1] Musick, S. and R. Malhotra, "Chasing the Elusive Sensor Manager", National Aerospace and Electronics Conference (NAECON), May, 1994

[2] K. J. Hintz and E.S. McVey, "Multi-Process Constrained Estimation", IEEE Transactions on Systems, Man, and Cybernetics, Vol. 21, No. 1, pp 237-244, Jan.,1991

[3] Llinas, J., J Hintz, and G. Beale. Applications of Automatic Control Theory to Sensor Scheduling. Technical Report # NAWCADWAR-93002-50, Naval Air Warfare Center Aircraft Division, Warminster, PA. September, 1992

[4] Kastella, K., "Discrimination Gain to Optimize Detection and Classification", to appear in IEEE Trans. on Systems, Man, and Cybernetics, 1996

[5] Schmaedeke, W., "Information-Based Sensor Management", SPIE Proceedings Vol. 1955, Signal

Processing, Sensor Fusion, and Target Recognition II, April, 1994.

[6] Popoli, R., "The Sensor Management Imperative", in Multitarget-Multi-sensor Tracking: Advanced Applications, Vol II, Y. Bar-Shalom, Ed., Artech House, 1992

[7] Sutton, R., A. Barto, and R. Williams. "Reinforcement Learning is Direct Adaptive Optimal Control", Control Systems, pp. 19-23, Vol. 12, No. 2, April, 1992

[8] Bertsekas, D.P. (1987). Dynamic Programming: Deterministic and Stochastic Models. Prentice-Hall Englewood Cliffs, NJ.

[9] Denton, R., E. Alcaraz, J. Knopow, J. Llinas, and K. Hintz. "Towards Modern Sensor Management Systems", Proceedings of the Sixth Joint Service Data Fusion Symposium, 14-18 Jun, 1993.

[10] Musick, S., "Sensor Management for Tactical Aircraft", Briefing given to Air Force Office of Scientific Research, Jul, 1994.

[11] Waltz, E. and J. Llinas. Multisensor Data Fusion, 1990, Artech House, Boston, MA.

[12] R.S. Sutton, "Learning to Predict by the Methods of Temporal Differences," *Machine Learning*, vol. 3, pp. 9-44, 1988

[13] Watkins, C.J.C.H., "Learning from Delayed Rewards", Ph.D. Thesis, Cambridge University, England, 1989

[14] Bertsekas, Dimitri P., "A Counterexample to Temporal Difference Learning", *Neural Computation*, vol. 7, pp 270-279 1995

[15] Jaakkola, T., M. Jordan, S. Singh, "On the Convergence of Stochastic Iterative Dynamic Programming Algorithms", *Neural Computation*, vol. 6, pp1185-1201, 1994

[16] Tesauro, G., "Practical Issues in Temporal Difference Learning", *Machine Learning*, vol.8, pp257-277, 1992

[17] Klopf, A. H., "A Neuronal Model of Classical Conditioning", *Psychobiology*, vol.16, pp85-125 1988